IJCCS (Indonesian Journal of Computing and Cybernetics Systems)
Vol.12, No.1, January 2018, pp. 83~94
ISSN (print): 1978-1520, ISSN (online): 2460-7258
DOI: 10.22146/ijccs.31144

Convolutional Neural Networks for Handwritten Javanese Character Recognition

Chandra Kusuma Dewa*[1], Amanda Lailatul Fadhilah[2], Afiahayati[3]
[1,2]Department of Informatics, Universitas Islam Indonesia, Yogyakarta, Indonesia
[3]Department of Computer Science and Electronics, FMIPA, UGM, Yogyakarta, Indonesia
e-mail: ***[1]chandra.kusuma@uii.ac.id**, [1]13523221@students.uii.ac.id, [2]afia@ugm.ac.id

Abstrak

Convolutional neural network (CNN) merupakan model baru di bidang pengenalan objek. Dikhususkan untuk input data yang bertipe spatial, CNN memiliki layer khusus, yaitu layer konvolusi dan layer pooling yang memungkinkan proses pembelajaran fitur secara hierarki dari data. Untuk pengenalan karakter tulisan tangan secara offline, seperti pengenalan karakter pada database MNIST, CNN menunjukkan performa yang lebih baik jika dibandingkan dengan model ataupun metode yang lain. Dengan memanfaatkan keunggulan CNN tersebut, dalam penelitian ini telah dikembangkan sebuah perangkat lunak dengan fitur pengolahan citra dan modul CNN untuk pengenalan karakter tulisan tangan Aksara Jawa. Perangkat lunak yang dikembangkan memanfaatkan deteksi kontur dan deteksi tepi Canny menggunakan pustaka OpenCV terhadap citra karakter Aksara Jawa untuk proses segmentasi. Modul CNN selanjutnya melakukan proses klasifikasi terhadap citra yang telah disegmentasi ke dalam 20 kelas. Untuk evaluasi, kinerja CNN dibandingkan dengan kinerja dari model Multilayer Perceptron (MLP) dari sisi akurasi klasifikasi dan waktu latih. Hasil pengujian menunjukkan akurasi dari model CNN mampu mengungguli akurasi dari model MLP meskipun CNN membutuhkan waktu latih yang lebih lama dibandingkan dengan MLP.

Kata kunci—*convolutional neural network, pengenalan karakter tulisan tangan, pengenalan Aksara Jawa*

Abstract

Convolutional neural network (CNN) is state-of-the-art method in object recognition task. Specialized for spatial input data type, CNN has special convolutional and pooling layers which enable hierarchical feature learning from the input space. For offline handwritten character recognition problem such as classifying character in MNIST database, CNN shows better classification result than any other methods. By leveraging the advantages of CNN over character recognition task, in this paper we developed a software which utilizes digital image processing methods and a CNN module for offline handwritten Javanese character recognition. The software performs image segmentation process using contour and Canny edge detection with OpenCV library over a captured handwritten Javanese character image. CNN will classify the segmented image into 20 classes of Javanese letters. For evaluation purposes, we compared CNN to multilayer perceptron (MLP) on classification accuracy and training time. Experiment results show that CNN model testing accuracy outperforms MLP accuracy although CNN needs more training time than MLP.

Keywords—*convolutional neural network, handwritten character recognition, Javanese character recognition*

Received December 11th,2017; Revised December 12th, 2017; Accepted December 13th, 2017

1. INTRODUCTION

Like many other ethnic groups in Indonesia, Javanese tribe also has a traditional script called *Aksara Jawa* or *Hanacaraka* which was intended as the font to write any documents in Javanese Language. *Aksara Jawa* consists of twenty base symbols or base characters which are usually called as *nglegéna* or *carakan*. Figure 1 shows twenty base characters of *Aksara Jawa* with their transcription in Roman alphabet.

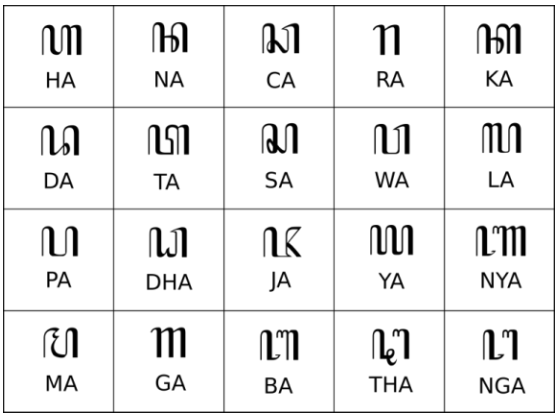


Figure 1 Javanese script characters

Unfortunately, Javanese people these days do not use *Aksara Jawa* in their daily life anymore. Although *Aksara Jawa* is taught in primary and secondary schools as part of *muatan-lokal* curriculum in Central Jawa and Special District of Yogyakarta which are the main base of Javanese culture, it mostly can only be seen in road name signs as the transcription for the Roman alphabet. In order to preserve Javanese culture, especially traditional Javanese script, not only does *Aksara Jawa* can be used in education but it also can be used as a means for daily communication [1]. To support the preservation of *Aksara Jawa* both in education and also in its usage in daily communication, a tool or a software which has an ability to automatically recognize handwritten Javanese character is needed.

There have been several studies of research regarding handwritten Javanese character recognition. Most of these studies use machine learning techniques such as hidden Markov model (HMM) or support vector machine (SVM) to perform the classification task with several feature extraction techniques. Widiarti and Wastu [2] used HMM to classify horizontal and vertical vector features while Nurul et. al. [3] used multi class SVM to recognize directional element feature from handwritten Javanese character dataset.

Some other studies of research use artificial neural networks to classify handwritten Javanese character. Isnawati [4] employed backpropagation neural networks and applied thinning method to handwritten Javanese character dataset. Wibowo et. al. [5] used multilayer perceptron (MLP) model which was trained using backpropagation algorithm. Similarly, Arum [6] used the combination of wavelet feature extraction technique and also backpropagation neural networks. Budhi and Adipranata [7] employed several artificial neural network methods with ICZ-ZCZ features for handwritten Javanese character recognition. Several other studies [8, 9, 10, 11] show that neural networks are able to perform image classification task with good performance if they are combined with appropriate feature extraction techniques.

In this study, we propose a software which employs convolutional neural networks (CNN) model with image processing module using OpenCV library to perform handwritten Javanese character recognition. CNN is one of deep learning methods which has special layers

which are able to perform feature extraction learning and extraction directly from raw input space. Currently, CNN becomes state-of-the-art method in image classification tasks.

2. METHODS

The objective of this study is to build a software which performs handwritten Javanese character recognition. In order to achieve the objective, we developed a classification module which employs a CNN model. We trained the CNN model with a dataset of handwritten Javanese character images. For evaluation purposes, we also used the same dataset to train an MLP model with one hidden layer and an MLP model with two hidden layers. We compared those three models in both classification accuracy and training time.

In this section, details of research methods used in this paper is described. The methodology consists of a) data acquisition for handwritten Javanese characters, b) building the CNN model, c) model training and model testing, and d) developing a web based application for handwritten Javanese character classification. Each part of the methodology will be described in the following subsections.

2.1 Data acquisition

In order to train the CNN model, an image dataset which consists of handwritten Javanese characters are needed. Unfortunately, there is no available public secondary dataset for handwritten Javanese characters. Therefore, in this study, a primary dataset of handwritten Javanese characters was manually collected from some people as shown in Figure 2. The dataset consists of 2,000 data which belongs to 20 classes of basic Javanese characters. Each class of Javanese character consists of 100 training data. Each data in the dataset is a 8-bit grayscale image which has a dimension of 28 x 28 pixels.

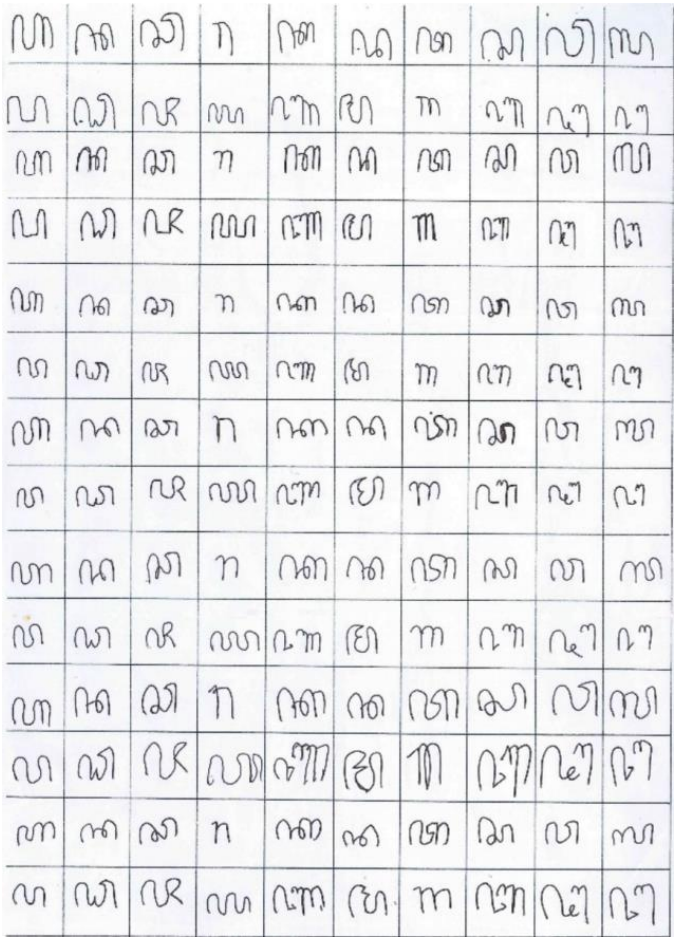


Figure 2 Handwritten Javanese chacter dataset

2. 2 Building CNN model

In essence, CNN can be seen as an extension for traditional neural network models, such as multilayer perceptron (MLP). A CNN model architecture consists of special layers to extract features from raw input space and a fully-connected neural network model with logistic regression classifier. The features which are usually called as feature maps are obtained from those special layers and then become inputs for fully-connected neural network which is actually an MLP model. Figure 3 shows an example of a CNN model architecture taken from Sermanet et. al. [12].

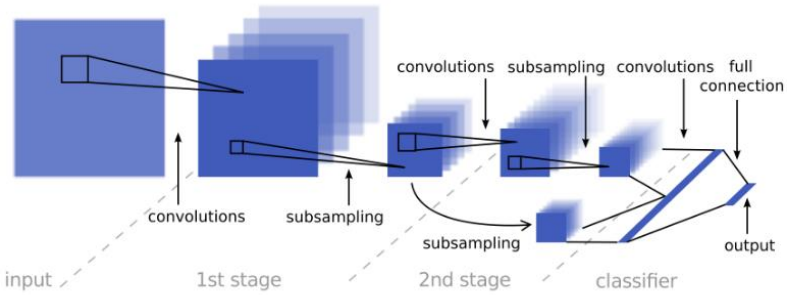


Figure 3 An example of CNN architecture [12]

From Figure 3, it can be seen that prior to be processed in the classifier, the input for CNN will be processed in two stages. Both stages consist of convolution and subsampling operations. It can also be seen that both convolution and subsampling operations will reduce dimensions of the inputs. A convolution operation will transform a single two-dimensional input matrix into some smaller two-dimentional matrix or feature maps. At the end of the second stage, a function is performed to flatten or transform each feature map from two-dimentional matrix to one-dimentional matrix so that the feature maps are ready to be classified with MLP or fully-connected neural network.

In CNN, the main objective of convolution and subsampling operations is for extracting features from raw input data. In order to achieve this objective, convolution operations which are multiplications of small kernel matrices and specified areas of a two-dimentional input matrix are performed. To produce a single smaller dimensions of feature map from an input matrix, the kernel will be shifted and several multiplications will be performed from left to right and from top to bottom over specified areas of the input matrix. The equation for a convolution operation to is defined in Formula (1) as stated in [13] as follows:

$$Q_j = f\left(\sum_{i=1}^{N} I_{i,i} * K_{i,j} + B_j\right), \tag{1}$$

where Q_j is an element of a single output matrix from a convolution operation. The output matrix is produced from an activation function f. First, the sum of all multiplications of kernel matrix $K_{i,j}$ and input matrix $I_{i,i}$ is computed, subsequently the bias value B_j is added to the elements of the resulting matrix. Finally, it becomes the input for function f. In this study, the activation function used is rectified linear unit (ReLU) which is defined in Formula (2).

$$f(x) = \begin{cases} x & (x \geq 0) \\ 0 & (x < 0). \end{cases} \tag{2}$$

After convolution operations, a subsampling or pooling operation will be applied to each of feature map for dimension reduction. In this study, the function for subsampling used is max pooling function so that prominent features can be obtained. To reduce the dimension of a single feature map, a two-dimentional m x n kernel will select the highest value of (m x n) neighbouring elements and produce one single element in a new feature map matrix. Similar to convolution operation, the kernel will also be shifted from left to right and from top to bottom to produce a new feature map.

To prevent overfitting and improve the performance of CNN model, a dropout regularization algorithm [14] also be applied in the training phase of the model. Using dropout algorithm, some neurons in CNN layers will be randomly disabled with Bernoulli distribution. Subsequently, in the testing phase, all of neurons in all layers in CNN model will be activated again. Srivastava et. al. [15] stated that dropout algorithm can improve the performance of neural networks in various benchmark datasets.

After several convolution and subsampling operations, feature maps will be flatten so that they are ready to be classified with MLP or fully-connected neural network. An MLP or a fully-connected neural network consists of several layers. Each layer consists of several neurons that will perform a matrix multiplication between an input matrix x_i and internal weights $w_{j,i}$ as defined in Formula (3) as follows:

$$u_j = f(\sum_{i=1}^{n} w_{j,i}x_i + b_j), \tag{3}$$

where b_j is bias value, n is the number of neurons in a single layer, and f is an activation function, such as ReLU function that has previously defined in Formula (2).

After processed in several layers, feature maps will be processed in the output layer. The output layer of a fully-connected neural network or an MLP is a softmax function that produces probabilities of classes $p(x)$ that the CNN input may belong. A softmax function is defined in Formula (4).

$$p(x) = \frac{e^x}{\sum_{k=1}^{K} e^x}. \tag{4}$$

2. 3 Model training and testing

A CNN model is included as one of neural network models which use supervised learning algorithm. This means that to update internal weight matrices in training phase, the model uses a cost function which calculate the distance between the output of the model, which is the predicted class, and the actual class that the input belongs. CNN models use cross entropy error function E as the cost function which is defined in Formula (5) as follows:

$$E = -\sum_{i=1}^{n} (t_i \log(x_i) + (1 - t_i) \log(1 - x_i)), \tag{5}$$

where t_i is the target class and x_i is the output of CNN model.

To perform training and testing phases of CNN model, the handwritten Javanese characters dataset was divided into 80% of training dataset and 20% of testing dataset with k-fold cross-validation technique. Xavier weight initialization [16] was used in each training fold to initialize internal weight matrices in the CNN model. In the testing phase, the performance of

each fold in CNN model was measured using a confusion matrix and the value of the classification accuracy. The classification accuracy is defined in Formula (6) as follows:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}, \quad (6)$$

where TP is true positive, TN is true negative, FP is false positive, and FN is false negative.

For this study, details of the CNN model architecture is shown in Table 1. The CNN model consists of three stages of convolution and subsampling operations, a fully-connected layer, and a softmax output layer. ReLU activation function and dropout regularization were applied to all layers in CNN model. The CNN model was built with Theano library, a deep learning library for Python. The CPU used was Intel Core i5-5200U, and the GPU was Nvidia GT940M.

Table 1 The CNN architecture used in this study

Layer Type	Size	Output Shape
Input	(1, 28, 28)	-
Convolution + ReLU	32 (3 x 3) filters	(32, 26, 26)
Max Pooling + Dropout	(2 x 2) filters	(32, 13, 13)
Convolution + ReLU	64 (2 x 2) filters	(64, 12, 12)
Max Pooling + Dropout	(2 x 2) filters	(64, 6, 6)
Convolution + ReLU	128 (3 x 3) filters	(128, 4, 4)
Max Pooling + Dropout	(2 x 2) filters	(128, 2, 2)
Fully-Connected + ReLU + Dropout	1,000 neurons	20
Softmax	20 way	20

We also trained and tested an MLP model with one hidden layer and an MLP model with two hidden layers with same handwritten Javanese character dataset. Accuracies of MLP models and CNN model were compared to verify whether convolution and subsampling layers in CNN model are able to learn features from the dataset. We compared the training time needed for both CNN and MLP model. Details of MLP models architecture are shown in Table 2 and Table 3 respectively.

Table 2 Architecture of MLP model with one hidden layer

Layer Type	Size	Output Shape
Input	784 neurons	-
Fully-Connected + ReLU	1,000 neurons	20
Softmax	20 way	20

It can be seen from Table 2 that the MLP model with one hidden layer has an input layer with 784 neurons and a hidden layer with 1,000 neurons. Each neuron in the input layer will receive a single pixel value from a handwritten Javanese character image which has a dimension of 28 x 28 pixels. The MLP model will produce 20 values of probability of classes which the input may belong.

Table 3 Architecture of MLP model with two hidden layers

Layer Type	Size	Output Shape
Input	784 neurons	-
Fully-Connected + ReLU	1,000 neurons	2,000
Fully-Connected + ReLU	2,000 neurons	20
Softmax	20 way	20

From the architecture described in Table 3, it can be seen that we used 1,000 neurons in the first layer and 2,000 neurons in the second layer. We also used ReLU activation function in hidden layers and softmax function in the output layer for both MLP models.

2. 4 Developing the web based application

The software developed in this study is a Django web based application which has an ability to access webcam and receive an image from user with upload method. It has two main features, namely segmentation of captured handwritten Javanese character image with OpenCV library and classification of the segmented image with CNN module. Figure 4 depicts the detailed architecture of the developed software in this study.

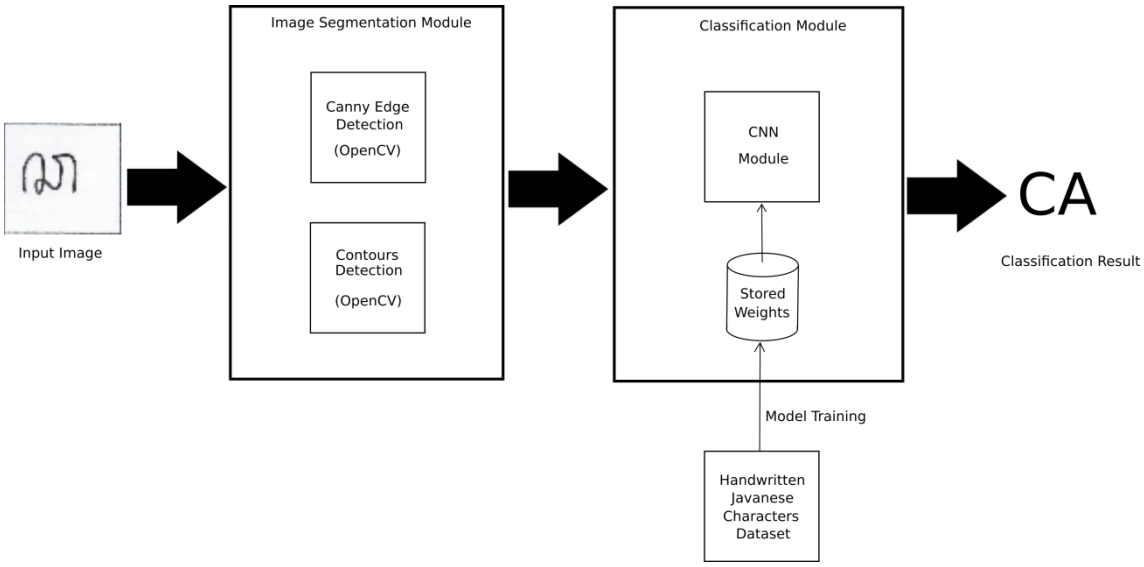


Figure 4. Architecture of the software developed in this study

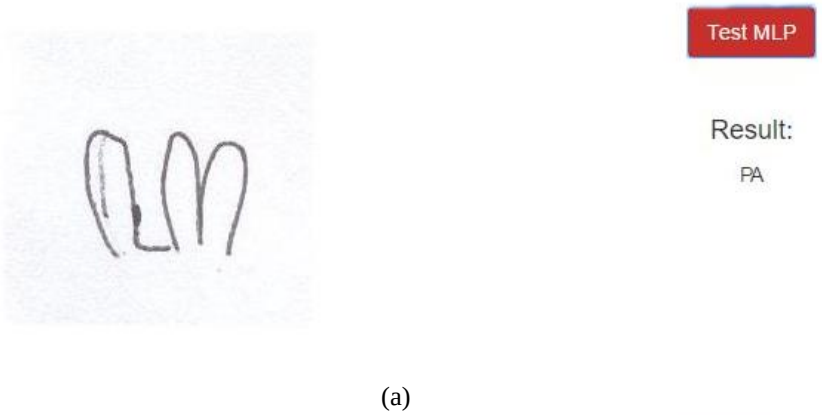
From Figure 4, it can be seen that the image segmentation module contains Canny edge detection and contours detection procedures with OpenCV library. The classification module consists of CNN module with stored weights which are obtained from model training using handwritten Javanese character dataset.

3. RESULTS AND DISCUSSION

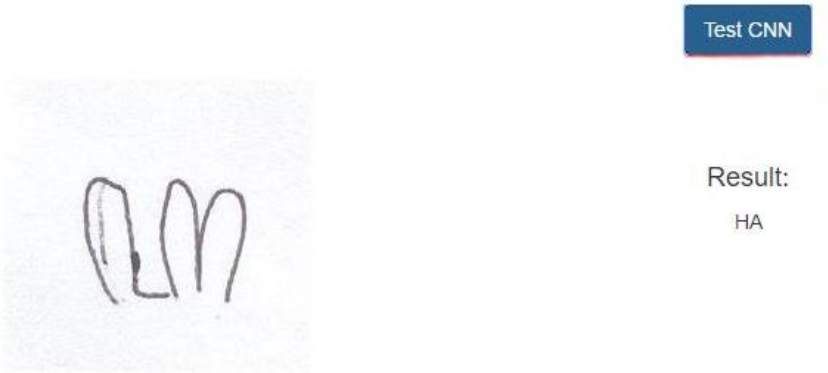
When a user first opens the URL of the software through a browser, both direct upload and webcam upload method can be used. The user can upload a captured handwritten Javanese character image or capture a handwritten Javanese character image using a webcam. After receiving the image, the software will use OpenCV library to perform segmentation procedure which its details will be described in the next subsection. Furthermore, the user can choose CNN method or MLP method to classify the segmented image. To illustrate how the classification process in the software works, Figure 5 depicts a state in the software which has already received a captured image and has already performed the classification task.

From Figure 5, it can be seen that the software could correctly recognize a character using CNN method, while it fails to do so with MLP method. Using MLP method, the character is incorrectly recognized as PA. On the other hand, CNN method can correctly recognize the character as HA. From this example, we can also see that the accuracy of CNN model is generally higher than the accuracy of MLP model. To verify this assumption, we conducted a

comparison procedure for testing accuracies for both CNN and MLP model which its details will be described in the next subsection, after the image segmentation procedure description.



(a)

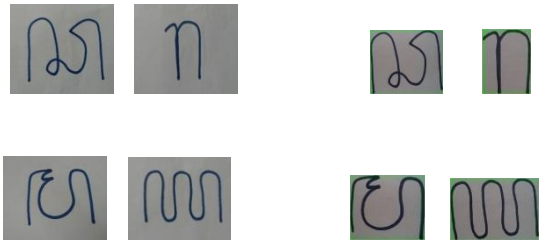


(b)

Figure 5 Character recognition results with (a) MLP model and (b) CNN model.

3.1 Image Segmentation Result

Prior to the classification procedure, a segmentation procedure will be performed for the captured handwritten Javanese character image. Canny edge detection algorithm and contours detection algorithm are applied to the captured image using OpenCV library to perform the segmentation procedure. This segmentation procedure will produce segmented images of Javanese letter characters. Figure 6 shows an example of the segmentation process as one of the main software features in this study.



(a) (b)

Figure 6 Image segmentation results, (a) before and (b) after.

3.2 Image Classification Result

We used two variables namely training time and testing accuracy using k-folds cross validation method to measure the performance of CNN for handwritten Javanese character recognition task. We also compared the training time and testing accuracy results to other results which are obtained from MLP with one hidden layer and MLP with two hidden layers. We used pixel values of the input image as inputs for MLP without prior feature extraction technique.

In each fold of the training phase for all models, we set a maximum epoch of 10,000 iterations. In each iteration, we calculated the cost function value and stored the internal weights matrix. Since the objective of the models is to minimize the cost function, we used an internal weights matrix of the iteration with the lowest cost function value for the models in the testing phase. Table 4 and Table 5 show the comparison results of those three models respectively.

Tabel 4 Training time result of the models

	Training Time				
Model	Cross Validation 1	Cross Validation 2	Cross Validation 3	Cross Validation 4	Cross Validation 5
MLP with one hidden layer	00:12:46	00:12:51	00:12:53	00:12:48	00:12:46
MLP with two hidden layer	00:34:51	00:34:49	00:34:41	00:34:56	00:34:57
CNN	01:48:30	01:49:37	01:50:19	01:49:59	01:50:10

From Table 4, we can see that MLP with one hidden layer requires minimal training time among other models for all cross validation folds. In average, MLP with one hidden layer only needs less than 13 minutes to train. If we add one more layer to the MLP, the training time will increase about 2.7 times. On the other hand, CNN requires more time to train compared to MLP due to the complex computations in convolution and subsampling layers. The model needs almost two hours for the training phase, which are about 4.6 longer than the training time for MLP model with one hidden layer.

Tabel 5 Testing accuracy results of the models

	Testing Accuracy				
Model	Cross Validation 1	Cross Validation 2	Cross Validation 3	Cross Validation 4	Cross Validation 5
MLP with one hidden layer	0.53	0.56	0.60	0.62	0.52
MLP with two hidden layer	0.44	0.49	0.52	0.59	0.47
CNN	0.82	0.85	0.89	0.89	0.78

Details of testing accuracy results of the three models are summarized in Table 5. Overall, classification accuracies of CNN model outperform classification accuracies of both MLP model. This means that convolutional and pooling layers in CNN model can successfully

learn features of the dataset. It can also be seen that adding more hidden layers to the MLP model does not improve its accuracy.

4. CONCLUSIONS

In this study, a software which employs CNN model to perform classification task for handwritten Javanese character recognition had been successfully developed. To quantify the performance of the classifier, k-folds cross validation technique had been used to measure the classification accuracy and the training time. The classification accuracy and the training time of CNN model were compared to the classification accuracy and the training time of MLP model with the same dataset. From the experiments, we conclude that CNN model's accuracy is better than MLP model's accuracy for handwritten Javanese character recognition task in all folds. However, CNN model needs longer time to be trained compared to MLP model. From the experiments, it can be seen that the accuracy of CNN model for the handwritten Javanese character dataset cannot reach 90% in all folds. This may be due to the insufficient number of the dataset since deep learning methods will give their best performance for huge amount of training data. CNN model optimization for a bigger handwritten Javanese character dataset is left for our future work.

ACKNOWLEDGMENTS

We would like to thank the Directorate of Research and Community Service, Universitas Islam Indonesia (DPPM-UII) for supporting this research financially with grant No. 021/Dir/DPPM/70/Pen.Pemula/PI/IV/2017. We also would like to thank Reyshahri Pradhana for helping us collecting the handwritten Javanese character dataset used in this study.

REFERENCES

[1] E. Nurhayati, Mulyana, H. Mulyani, and Suwardi, "Strategi pemertahanan Bahasa Jawa di Provinsi Daerah Istimewa Yogyakarta," *LITERA: Jurnal Penelitian Bahasa, Sastra, dan Pengajarannya*, vol. 12, no. 1, pp. 159-166, 2013. [Online]. Available: https://journal.uny.ac.id/index.php/litera/article/view/1338. [Accessed 12 August 2017].

[2] A. R. Widiarti and P. N. Wastu, "Javanese character recognition using hidden Markov model," *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, vol. 3, no. 9, pp. 2201-2204, 2009. [Online]. Available: http://waset.org/publications/10027/javanese-character-recognition-using-hidden-markov-model. [Accessed 2 January 2017].

[3] A. H. Nurul, M. D. Sulistiyo, and R. N. Dayawati, "Pengenalan Aksara Jawa tulisan tangan menggunakan directional element feature dan multi class support vector machine," in *Prosiding Konferensi Nasional Teknologi Informasi dan Aplikasinya*, 13 September 2014, Palembang, Indonesia [Online]. Available: http://seminar.ilkom.unsri.ac.id/index.php/kntia/article/view/733 /409. [Accessed: 4 February 2017].

[4] B. Isnawati, "Analisis implementasi jaringan syaraf tiruan backpropagation untuk

klasifikasi huruf dasar Aksara Jawa," Undergraduate Thesis, Department of Electrical Engineering and Information Technology, Universitas Gadjah Mada, Yogyakarta, 2015.

[5] M. C. Wibowo, I. D. G. R. Mardiana, and S. Wirakusuma, "Pengenalan pola tulisan tangan Aksara Jawa menggunakan multilayer perceptron," in *Prosiding Seminar Nasional Teknologi Informasi dan Multimedia*, 6-8 February 2015, Yogyakarta, Indonesia [Online]. Available: http://ojs.amikom.ac.id/index.php/semnasteknomedia/article/view/907. [Accessed: 10 March 2017].

[6] N. B. Arum, "Pengenalan pola Aksara Jawa nglegena berbasis wavelet dan jaringan syaraf tiruan backpropagation," Undergraduate Thesis, Department of Electrical Engineering and Information Technology, Universitas Gadjah Mada, Yogyakarta, 2016.

[7] G. S. Budhi and R. Adipranata, "Handwritten Javanese character recognition using several artificial neural network methods," *Journal of ICT Research and Applications*, vol. 8, no. 3, pp. 195-212, 2015. [Online]. Available: http://journals.itb.ac.id/index.php/jictra/article/view/769. [Accessed 5 July 2017].

[8] K. Syaban and A. Harjoko, "Klasifikasi varietas cabai berdasarkan morfologi daun menggunakan backpropagation neural network," *(IJCCS) Indonesian Journal of Computing and Cybernetics Systems*, vol. 10, no. 2, pp. 161-172, 2016. [Online]. Available: https://journal.ugm.ac.id/ijccs/article/view/16628. [Accessed 15 April 2017].

[9] Herman and A. Harjoko, "Pengenalan spesies gulma berdasarkan bentuk dan tekstur daun menggunakan jaringan syaraf tiruan," *(IJCCS) Indonesian Journal of Computing and Cybernetics Systems*, vol. 9, no. 2, pp. 207-218, 2015. [Online]. Available: https://journal.ugm.ac.id/ijccs/article/view/7549. [Accessed 11 February 2017].

[10] S. Aliaji and A. Harjoko, "Identifikasi barcode pada gambar yang ditangkap kamera digital menggunakan metode JST," *(IJCCS) Indonesian Journal of Computing and Cybernetics Systems*, vol. 7, no. 2, pp. 121-132, 2013. [Online]. Available: https://journal.ugm.ac.id/ijccs/article/view/3351. [Accessed 9 May 2017].

[11] S. Zubair and A. Solichin, "Pengenalan karakter sandi rumput pramuka menggunakan jaringan syaraf tiruan dengan metode backpropagation," in *Prosiding Seminar Nasional Teknologi Informasi dan Multimedia*, 4 February 2017, Yogyakarta, Indonesia [Online]. Available: http://ojs.amikom.ac.id/index.php/semnasteknomedia/article/view/1764. [Accessed: 10 August 2017].

[12] P. Sermanet, S. Chintala, and Y. LeCun, "Convolutional neural networks applied to house numbers digit classification," in *Proceedings of 2012 21st IEEE International Conference on Pattern Recognition (ICPR)*, 11-15 November 2012, Tsukuba, Japan [Online]. Available: http://ieeexplore.ieee.org/document/6460867/. [Accessed: 10 March 2017].

[13] Q. Li, W. Cai, X. Wang, Y. Zhou, D. D. Feng, and M. Chen, "Medical image classification with convolutional neural network," in *Proceedings of 2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)*, 10-12 December 2014, Singapore, Singapore [Online]. Available: http://ieeexplore.ieee.org/document/7064414/. [Accessed: 4 June 2017].

[14] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *Proceedings of 2013 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 26-31 May 2013,

Vancouver, Canada [Online]. Available: http://ieeexplore.ieee.org/document/6639346/. [Accessed: 7 June 2017].

[15] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929-1958, 2014. [Online]. Available: http://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf. [Accessed 5 May 2017].

[16] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of International Conference on Artificial Intelligence and Statistics*, 13-15 May 2010, Sardinia, Italy [Online]. Available: http://proceedings.mlr.press/v9/glorot10a/glorot10a.pdf. [Accessed: 11 July 2017].